Geometric Modeling

Project Ideas

Julian Panetta — fjp234@nyu.edu

Questions on HW4?

Project (HW6)

- **Proposals must be approved by 4/27/17** (Thursday after next)
- Implement a graphics paper from scratch (no provided code)
- Using libigl is fine, other dependencies must be pre-approved
- Bonus: do two projects for 1.5x points

Project Ideas

- Choose from **6 suggestions**
  - Harmonic Coordinates for Character Animation
  - Multiscale Biharmonic Kernels
  - Smooth Feature Lines on Surface Meshes
  - Interactive Geometry Remeshing
  - Generalized Winding Numbers
  - Transfusive Image Manipulation
- … **or propose your own**

Harmonic Coordinates

Harmonic Coordinates for Character Articulation

Pushkar Joshi   Mark Meyer   Tony DeRose
Brian Green   Tom Sanocki
Pixar Animation Studios

(a) (b) (c)

Figure 1: *A character posed using using harmonic coordinates. (a) The character and cage (shown in black) at bind-time; (b) and (c) are two poses from an animated clip. All images © Disney/Pixar.*

Abstract

In this paper we consider the problem of creating and controlling volume deformations used to articulate characters for use in high-end applications such as computer generated feature films. We introduce a method we call harmonic coordinates that significantly improves upon existing volume deformation techniques. Our deformations are controlled using a topologically flexible structure, called a cage, that consists of a closed three dimensional mesh. The cage can optionally be augmented with additional interior vertices, edges, and faces to more precisely control the interior behavior of the deformation. We show that harmonic coordinates are generalized barycentric coordinates that can be extended to any dimension. Moreover, they are the first system of generalized barycentric coordinates that are non-negative even in strongly concave situations, and their magnitude falls off with distance as measured within the cage.

CR Categories: I.3.5 [Computational Geometry and Object Modeling]: Geometric algorithms, languages, and systems.

Keywords: Barycentric coordinates, mean value coordinates, free form deformations, rigging.

1 Introduction

Character articulation, sometimes called rigging, is an important component of high-end animation systems of the kind used in feature film production. Modern high-end systems, most notably SoftImage XSI® and Maya®, offer a variety of articulation methods such as enveloping [Lewis et al. 2000], blend shapes [Joshi et al. 2006], and chains of arbitrary deformations. In the realm of deformations, free-form deformations as introduced by Sederberg and Parry [1986] are particularly popular for a number of reasons. First, they offer smooth and intuitive control over the motion of the character using only a few parameters, namely, the locations of the free-form lattice control points. Second, there are virtually no restrictions on the three-dimensional model of the character — the only requirement is that the character model is completely enclosed by the control lattice.

However, free-form deformation has some drawbacks. Articulating a multi-limbed character is best accomplished using a lattice that conforms to the geometry of the character. However, given the topological rigidity of a lattice, it is often necessary to combine several overlapping lattices, and each of the lattices possess interior points that can be difficult and annoying to articulate. The problem of multiple overlapping lattices was addressed by MacCracken and Joy [1996] where lattices were generalized to arbitrary volume meshes, but their method still requires the introduction and articulation of numerous interior control points.

Ju et al [2005] introduced a promising new approach that is even more topologically flexible, wherein the character to be deformed (henceforth called the *object*) is positioned relative to a coarse closed triangular surface mesh (henceforth called the *cage*). The object is then "bound" to the cage by computing a weight $g_i(p)$ of each cage vertex C_i evaluated at the position of every object point p. As the cage vertices are moved to new locations C_i', the deformed points p' are computed from

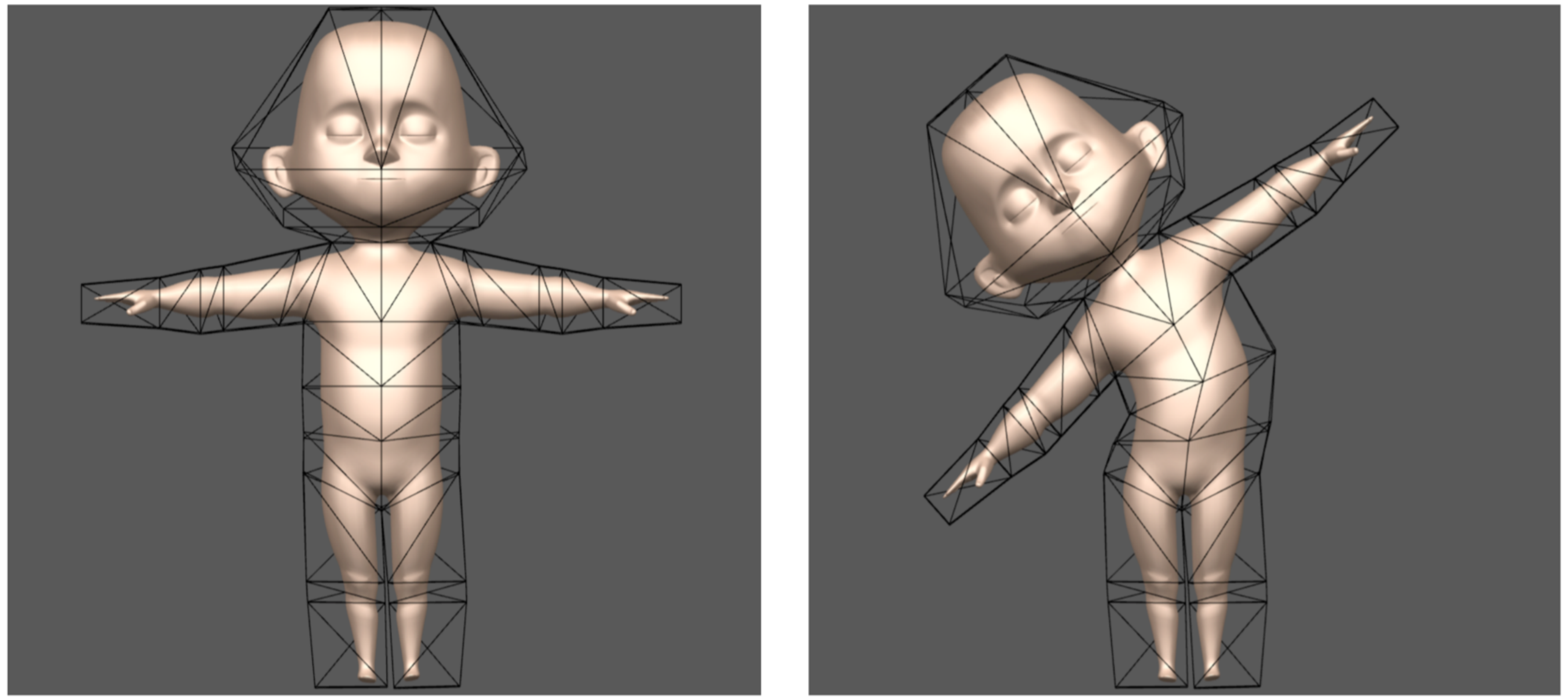
$$p' = \sum_i g_i(p) C_i'. \tag{1}$$

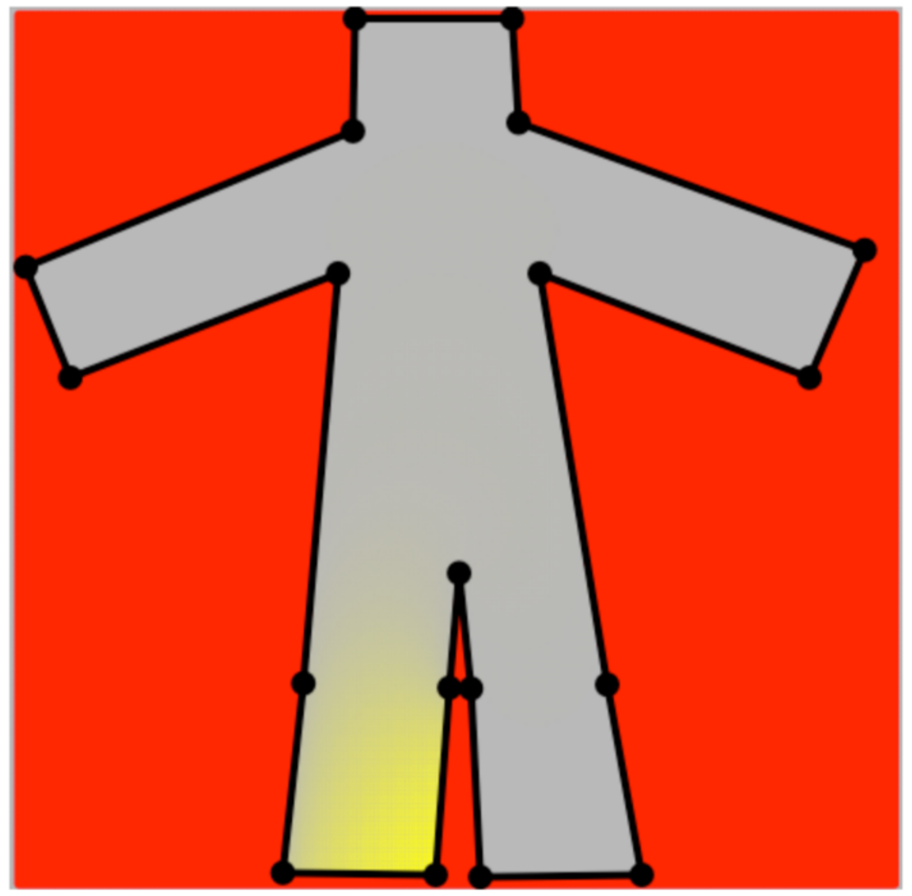
An example is shown in Figure 2(b). The weight functions $g_i(p)$ used by Ju et al. are known as mean value coordinates [Floater 2003; Floater et al. 2005; Ju et al. 2005]. Mean value coordinates are a form of generalized barycentric coordinates that have a number of uses, but they are particularly interesting in the context of character articulation because:

ACM Reference Format
Joshi, P., Meyer, M., DeRose, T., Green, B., Sanocki, T. 2007. Harmonic Coordinates for Character Articulation. *ACM Trans. Graph. 26*, 3, Article 71 (July 2007), 9 pages. DOI = 10.1145/1239451.1239522 http://doi.acm.org/10.1145/1239451.1239522.

NYU | COURANT

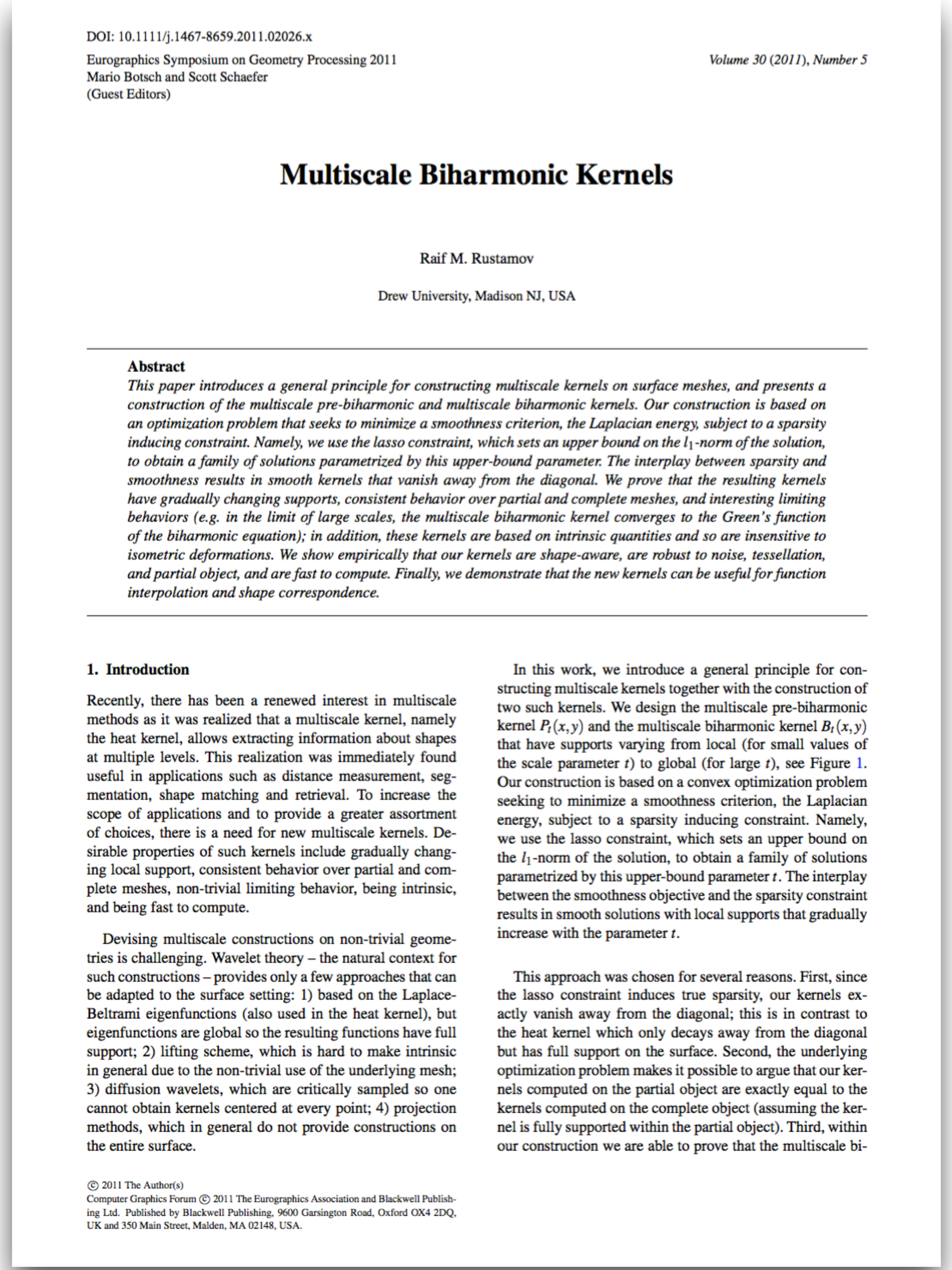
Harmonic Coordinates





- Draw a cage around an image
- Construct “coordinate functions” for each cage vertex
  - Smooth functions over the mesh
  - 1 on a particular cage vertex, 0 on all others
  - Compute them as solutions to Laplace equation!
- Use these coordinate functions to interpolate a cage deformation

Multiscale Biharmonic Kernels


DOI: 10.1111/j.1467-8659.2011.02026.x
Eurographics Symposium on Geometry Processing 2011
Mario Botsch and Scott Schaefer
(Guest Editors)
Volume 30 (2011), Number 5

Multiscale Biharmonic Kernels

Raif M. Rustamov

Drew University, Madison NJ, USA

Abstract
This paper introduces a general principle for constructing multiscale kernels on surface meshes, and presents a construction of the multiscale pre-biharmonic and multiscale biharmonic kernels. Our construction is based on an optimization problem that seeks to minimize a smoothness criterion, the Laplacian energy, subject to a sparsity inducing constraint. Namely, we use the lasso constraint, which sets an upper bound on the l_1-norm of the solution, to obtain a family of solutions parametrized by this upper-bound parameter. The interplay between sparsity and smoothness results in smooth kernels that vanish away from the diagonal. We prove that the resulting kernels have gradually changing supports, consistent behavior over partial and complete meshes, and interesting limiting behaviors (e.g. in the limit of large scales, the multiscale biharmonic kernel converges to the Green's function of the biharmonic equation); in addition, these kernels are based on intrinsic quantities and so are insensitive to isometric deformations. We show empirically that our kernels are shape-aware, are robust to noise, tessellation, and partial object, and are fast to compute. Finally, we demonstrate that the new kernels can be useful for function interpolation and shape correspondence.

1. Introduction

Recently, there has been a renewed interest in multiscale methods as it was realized that a multiscale kernel, namely the heat kernel, allows extracting information about shapes at multiple levels. This realization was immediately found useful in applications such as distance measurement, segmentation, shape matching and retrieval. To increase the scope of applications and to provide a greater assortment of choices, there is a need for new multiscale kernels. Desirable properties of such kernels include gradually changing local support, consistent behavior over partial and complete meshes, non-trivial limiting behavior, being intrinsic, and being fast to compute.

Devising multiscale constructions on non-trivial geometries is challenging. Wavelet theory – the natural context for such constructions – provides only a few approaches that can be adapted to the surface setting: 1) based on the Laplace-Beltrami eigenfunctions (also used in the heat kernel), but eigenfunctions are global so the resulting functions have full support; 2) lifting scheme, which is hard to make intrinsic in general due to the non-trivial use of the underlying mesh; 3) diffusion wavelets, which are critically sampled so one cannot obtain kernels centered at every point; 4) projection methods, which in general do not provide constructions on the entire surface.

In this work, we introduce a general principle for constructing multiscale kernels together with the construction of two such kernels. We design the multiscale pre-biharmonic kernel $P_t(x,y)$ and the multiscale biharmonic kernel $B_t(x,y)$ that have supports varying from local (for small values of the scale parameter t) to global (for large t), see Figure 1. Our construction is based on a convex optimization problem seeking to minimize a smoothness criterion, the Laplacian energy, subject to a sparsity inducing constraint. Namely, we use the lasso constraint, which sets an upper bound on the l_1-norm of the solution, to obtain a family of solutions parametrized by this upper-bound parameter t. The interplay between the smoothness objective and the sparsity constraint results in smooth solutions with local supports that gradually increase with the parameter t.

This approach was chosen for several reasons. First, since the lasso constraint induces true sparsity, our kernels exactly vanish away from the diagonal; this is in contrast to the heat kernel which only decays away from the diagonal but has full support on the surface. Second, the underlying optimization problem makes it possible to argue that our kernels computed on the partial object are exactly equal to the kernels computed on the complete object (assuming the kernel is fully supported within the partial object). Third, within our construction we are able to prove that the multiscale bi-

NYU | COURANT

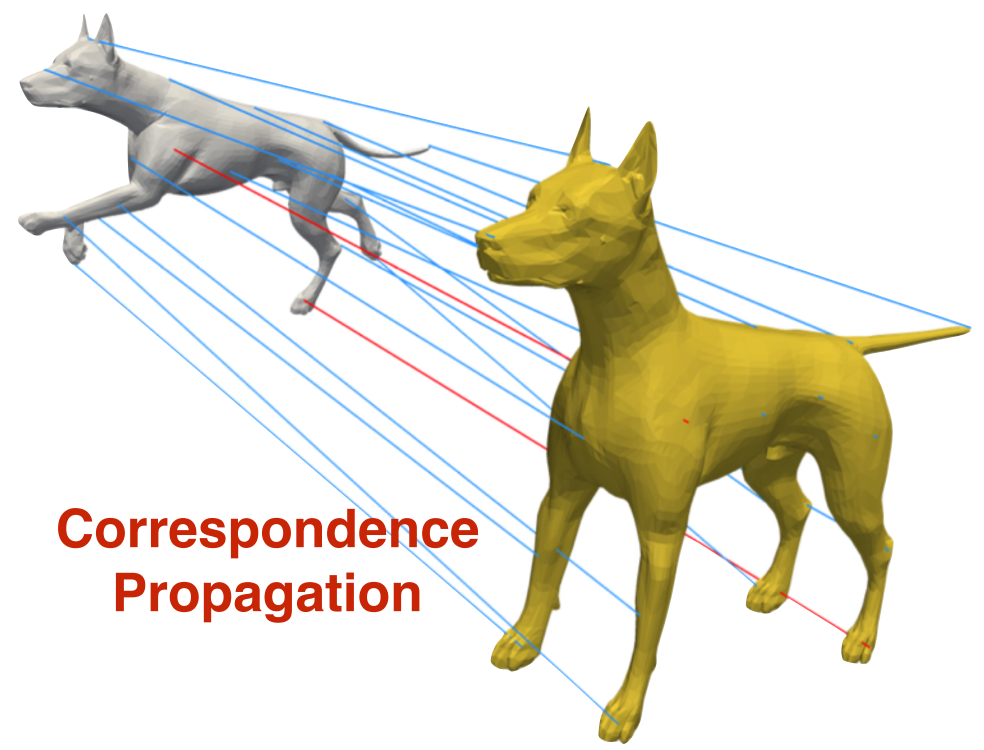
Multiscale Biharmonic Kernels

$t = t_{max}/32$ $t = t_{max}/16$ $t = t_{max}/8$ $t = t_{max}/4$ $t = t_{max}/2$ $t = t_{max}$

- Construct **smooth** basis functions centered at a point "y" that **vanish** outside radius controlled by "t":

$$\min_f \int_S (\Delta f)^2 \quad \text{subject to} \int_S |f| \leq t \quad \text{and} \quad \begin{array}{l} f(y) = 1 \\ \int_S f = 0 \end{array}$$

- Applications: RBF interpolation on surface,
  Correspondence propagation



NYU | COURANT

Smooth Feature Lines on Surface Meshes

Eurographics Symposium on Geometry Processing (2005)
M. Desbrun, H. Pottmann (Editors)

Smooth Feature Lines on Surface Meshes

Klaus Hildebrandt † Konrad Polthier † Max Wardetzky †

Zuse Institute Berlin

Abstract
Feature lines are salient surface characteristics. Their definition involves third and fourth order surface derivatives. This often yields to unpleasantly rough and squiggly feature lines since third order derivatives are highly sensitive against unwanted surface noise. The present work proposes two novel concepts for a more stable algorithm producing visually more pleasing feature lines: First, a new computation scheme based on discrete differential geometry is presented, avoiding costly computations of higher order approximating surfaces. Secondly, this scheme is augmented by a filtering method for higher order surface derivatives to improve both the stability of the extraction of feature lines and the smoothness of their appearance.

1. Introduction

Feature lines are curves on surfaces carrying - *in a few strokes* - visually most prominent characteristics. Their extraction from discrete meshes has become an area of intense research [Thi96] [OBS04] [YBS05] [SF03] [HG01] [CP05b] with applications ranging from structure analysis in medical data [MAM97] [Sty03] over non photorealistic rendering techniques [IFP95] to surface segmentation [SF04].

Mathematically, feature lines are described as local extrema of principal curvatures along corresponding principal directions. On smooth surfaces, these extrema have been subject to intense mathematical studies based on techniques from differential geometry, singularity theory and bifurcation theory [Por94] [Koe90] [BAK97] [CP05a].

Reliable computations of discrete curvature measures on meshes are key to many methods in geometric modeling and computer graphics. In particular, the detection of ridges and features requires the computation of first and even second order derivatives of principal curvatures. Higher order derivatives, in turn, are not a straightforward concept on discrete surfaces, due to their piecewise linear nature. Consequently, the standard approach has been to locally (or sometimes globally) fit a smooth (often polynomial) surface to the vertex coordinates and to then compute curvatures from this smooth surface, see e.g. [CP03] [GI04] [OBS04].

In contrast, our methodology is based on utilizing discrete differential operators on piecewise linear meshes, an approach which avoids often costly preprocessing steps such as higher order surface-fitting techniques or implicit surface schemes. Discretizing smooth geometric quantities has been found to be a powerful numerical machinery for geometry processing: The discrete mesh Laplacian [PP93] [MDSB03] is utilized for isotropic remeshing [AdVDI03], isotropic denoising [DMSB99], and mesh parameterization [GY03]. A model of a purely discrete shape operator [CSM03] [HP04] has been successfully employed in anisotropic remeshing and smoothing schemes [ACSD*03] [HP04] as well as thin shell simulations [GHDS03], to name a few of its most prominent applications. In the present work, we use the discrete approach for both extracting feature lines and smoothing thereof.

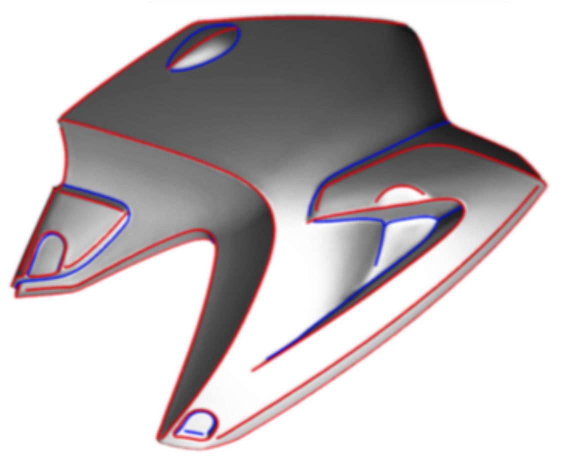
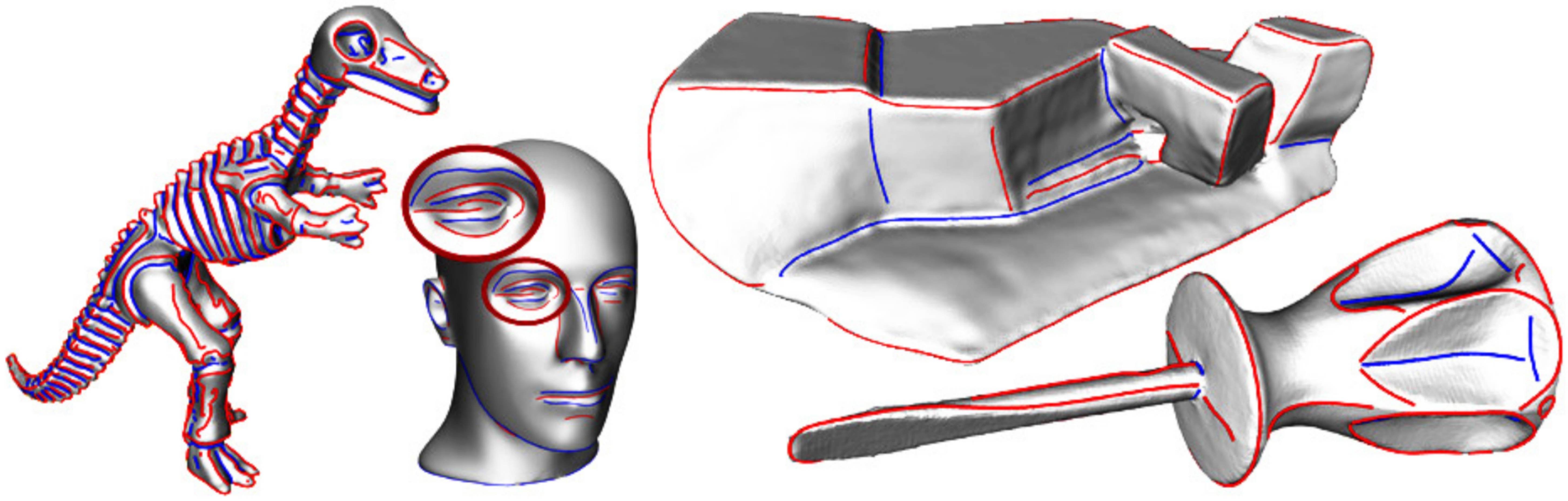


Figure 1: *Smooth feature lines on a motorcycle body part.*

† Supported by the DFG Research Center MATHEON "Mathematics for key technologies" in Berlin.

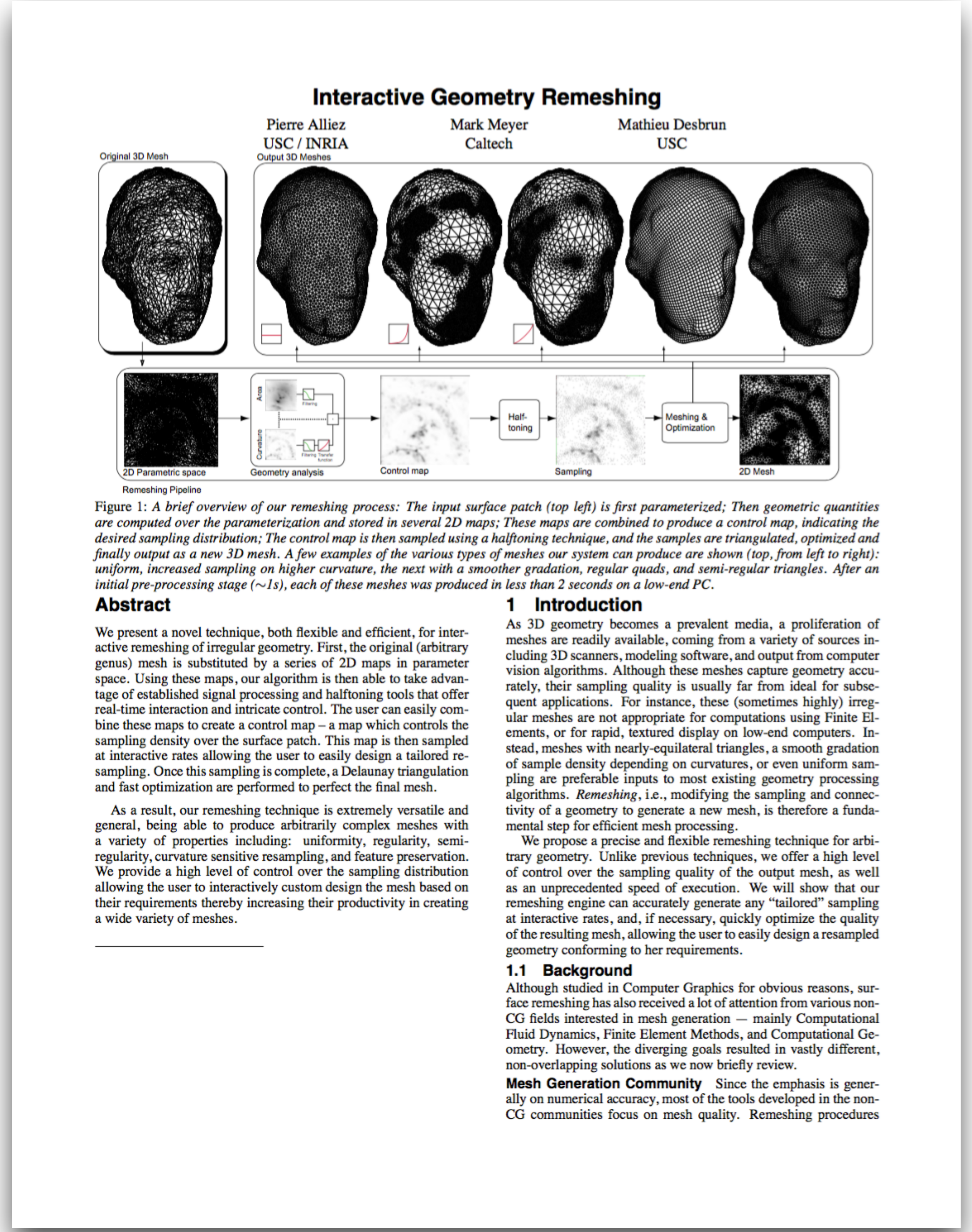
Smooth Feature Lines on Surface Meshes



- Detect "salient ridges" on meshes
- Compute points with extreme principal curvatures
  - Under certain conditions, these form feature curves
  - Checking these conditions requires computing 1st and 2nd derivatives of curvature

NYU COURANT

Interactive Geometry Remeshing



Interactive Geometry Remeshing

Pierre Alliez
USC / INRIA

Mark Meyer
Caltech

Mathieu Desbrun
USC

Figure 1: *A brief overview of our remeshing process: The input surface patch (top left) is first parameterized; Then geometric quantities are computed over the parameterization and stored in several 2D maps; These maps are combined to produce a control map, indicating the desired sampling distribution; The control map is then sampled using a halftoning technique, and the samples are triangulated, optimized and finally output as a new 3D mesh. A few examples of the various types of meshes our system can produce are shown (top, from left to right): uniform, increased sampling on higher curvature, the next with a smoother gradation, regular quads, and semi-regular triangles. After an initial pre-processing stage (∼1s), each of these meshes was produced in less than 2 seconds on a low-end PC.*

Abstract

We present a novel technique, both flexible and efficient, for interactive remeshing of irregular geometry. First, the original (arbitrary genus) mesh is substituted by a series of 2D maps in parameter space. Using these maps, our algorithm is then able to take advantage of established signal processing and halftoning tools that offer real-time interaction and intricate control. The user can easily combine these maps to create a control map – a map which controls the sampling density over the surface patch. This map is then sampled at interactive rates allowing the user to easily design a tailored resampling. Once this sampling is complete, a Delaunay triangulation and fast optimization are performed to perfect the final mesh.

As a result, our remeshing technique is extremely versatile and general, being able to produce arbitrarily complex meshes with a variety of properties including: uniformity, regularity, semi-regularity, curvature sensitive resampling, and feature preservation. We provide a high level of control over the sampling distribution allowing the user to interactively custom design the mesh based on their requirements thereby increasing their productivity in creating a wide variety of meshes.

1 Introduction

As 3D geometry becomes a prevalent media, a proliferation of meshes are readily available, coming from a variety of sources including 3D scanners, modeling software, and output from computer vision algorithms. Although these meshes capture geometry accurately, their sampling quality is usually far from ideal for subsequent applications. For instance, these (sometimes highly) irregular meshes are not appropriate for computations using Finite Elements, or for rapid, textured display on low-end computers. Instead, meshes with nearly-equilateral triangles, a smooth gradation of sample density depending on curvatures, or even uniform sampling are preferable inputs to most existing geometry processing algorithms. *Remeshing*, i.e., modifying the sampling and connectivity of a geometry to generate a new mesh, is therefore a fundamental step for efficient mesh processing.

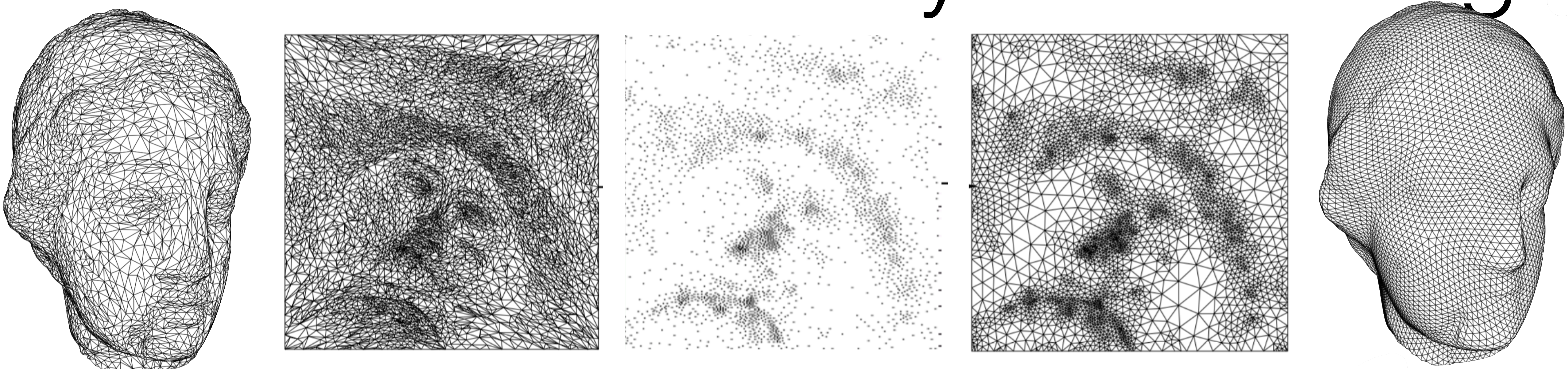
We propose a precise and flexible remeshing technique for arbitrary geometry. Unlike previous techniques, we offer a high level of control over the sampling quality of the output mesh, as well as an unprecedented speed of execution. We will show that our remeshing engine can accurately generate any “tailored” sampling at interactive rates, and, if necessary, quickly optimize the quality of the resulting mesh, allowing the user to easily design a resampled geometry conforming to her requirements.

1.1 Background

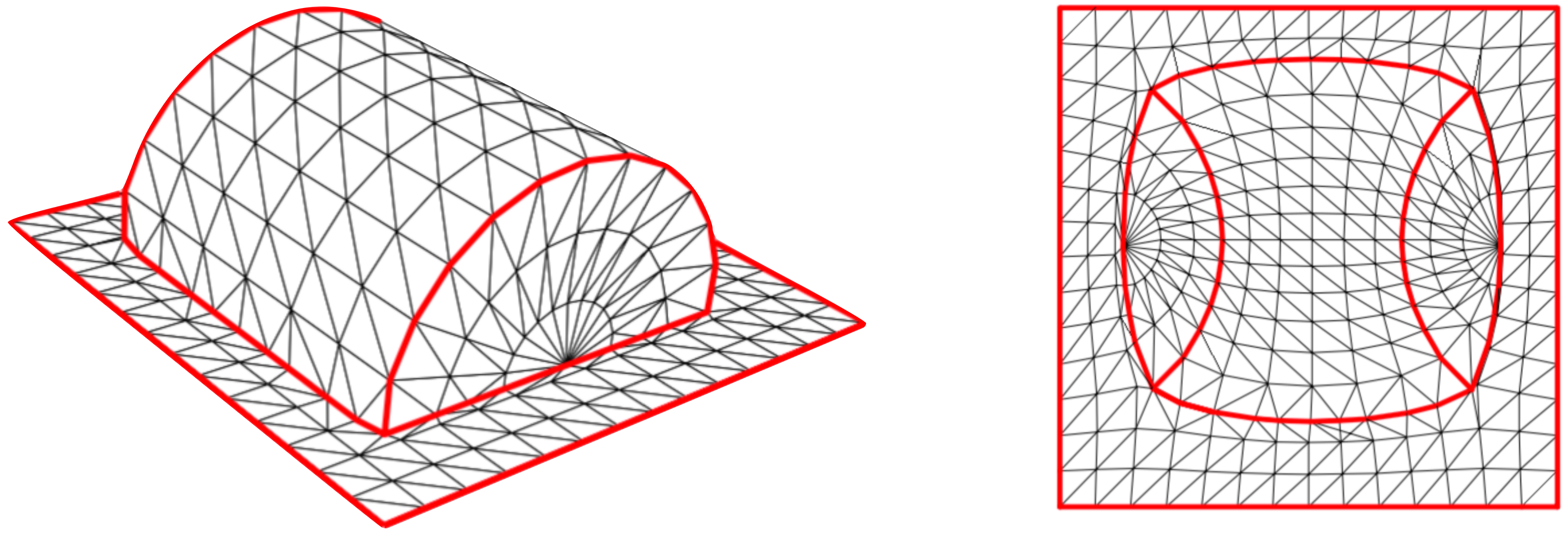
Although studied in Computer Graphics for obvious reasons, surface remeshing has also received a lot of attention from various non-CG fields interested in mesh generation — mainly Computational Fluid Dynamics, Finite Element Methods, and Computational Geometry. However, the diverging goals resulted in vastly different, non-overlapping solutions as we now briefly review.

Mesh Generation Community Since the emphasis is generally on numerical accuracy, most of the tools developed in the non-CG communities focus on mesh quality. Remeshing procedures

Interactive Geometry Remeshing

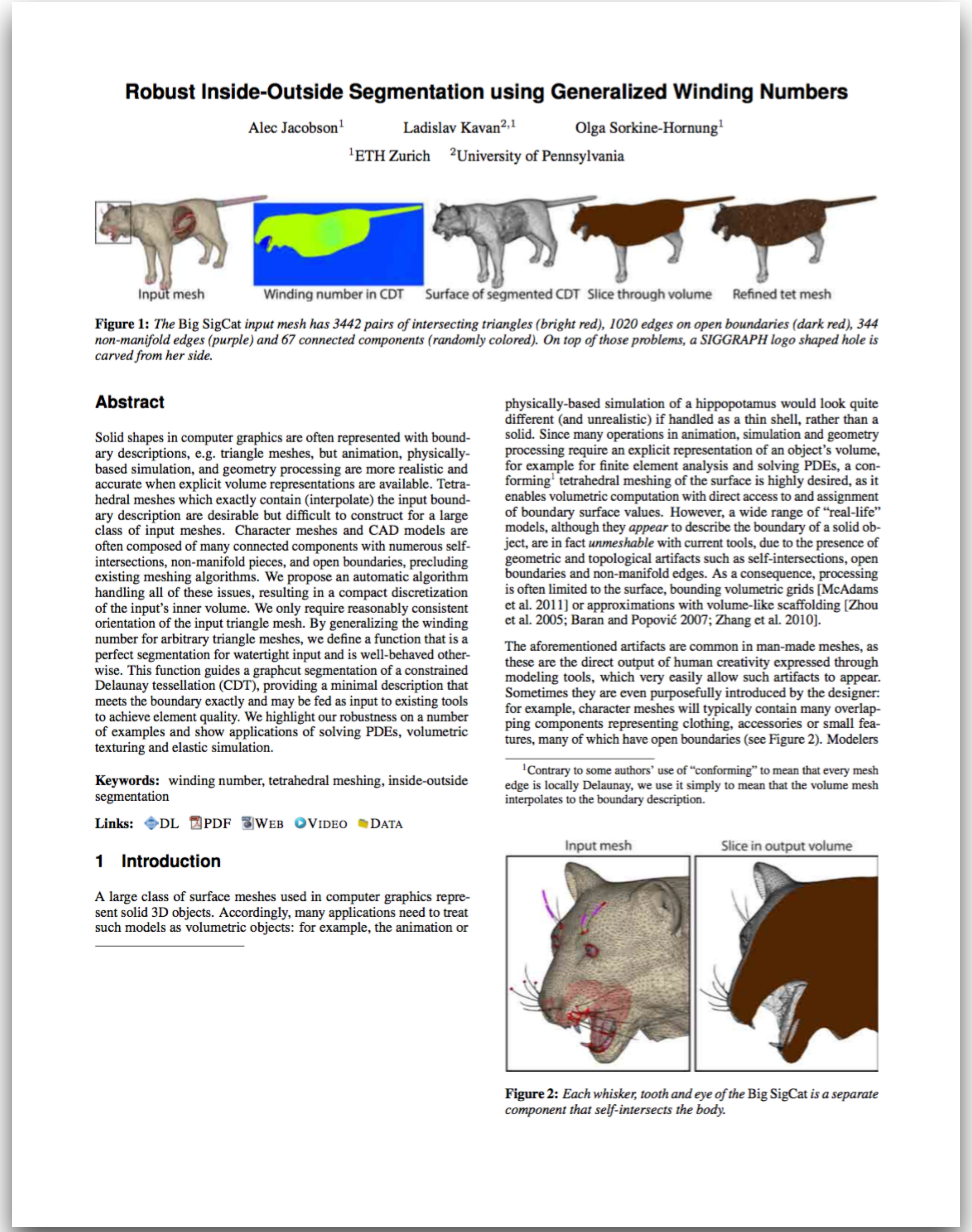


- **Parametrize mesh** into (UV) plane, **triangulate** UV plane nicely, **transfer mesh back to 3D**.
- Design plane triangulation to **minimize post-transfer distortion**
  - Compensate for parametrization's area distortion: **vertex density proportional to area scaling**.
  - Detect creases in original mesh, constrain these to be edges in triangulation.
  - Optionally increase sampling in "important regions," e.g. high curvature.



NYU | COURANT

Generalized Winding Numbers



Robust Inside-Outside Segmentation using Generalized Winding Numbers

Alec Jacobson[1] Ladislav Kavan[2,1] Olga Sorkine-Hornung[1]

[1]ETH Zurich [2]University of Pennsylvania

Input mesh | Winding number in CDT | Surface of segmented CDT | Slice through volume | Refined tet mesh

Figure 1: *The* Big SigCat *input mesh has 3442 pairs of intersecting triangles (bright red), 1020 edges on open boundaries (dark red), 344 non-manifold edges (purple) and 67 connected components (randomly colored). On top of those problems, a SIGGRAPH logo shaped hole is carved from her side.*

Abstract

Solid shapes in computer graphics are often represented with boundary descriptions, e.g. triangle meshes, but animation, physically-based simulation, and geometry processing are more realistic and accurate when explicit volume representations are available. Tetrahedral meshes which exactly contain (interpolate) the input boundary description are desirable but difficult to construct for a large class of input meshes. Character meshes and CAD models are often composed of many connected components with numerous self-intersections, non-manifold pieces, and open boundaries, precluding existing meshing algorithms. We propose an automatic algorithm handling all of these issues, resulting in a compact discretization of the input's inner volume. We only require reasonably consistent orientation of the input triangle mesh. By generalizing the winding number for arbitrary triangle meshes, we define a function that is a perfect segmentation for watertight input and is well-behaved otherwise. This function guides a graphcut segmentation of a constrained Delaunay tessellation (CDT), providing a minimal description that meets the boundary exactly and may be fed as input to existing tools to achieve element quality. We highlight our robustness on a number of examples and show applications of solving PDEs, volumetric texturing and elastic simulation.

Keywords: winding number, tetrahedral meshing, inside-outside segmentation

Links: DL PDF WEB VIDEO DATA

1 Introduction

A large class of surface meshes used in computer graphics represent solid 3D objects. Accordingly, many applications need to treat such models as volumetric objects: for example, the animation or physically-based simulation of a hippopotamus would look quite different (and unrealistic) if handled as a thin shell, rather than a solid. Since many operations in animation, simulation and geometry processing require an explicit representation of an object's volume, for example for finite element analysis and solving PDEs, a conforming[1] tetrahedral meshing of the surface is highly desired, as it enables volumetric computation with direct access to and assignment of boundary surface values. However, a wide range of "real-life" models, although they *appear* to describe the boundary of a solid object, are in fact *unmeshable* with current tools, due to the presence of geometric and topological artifacts such as self-intersections, open boundaries and non-manifold edges. As a consequence, processing is often limited to the surface, bounding volumetric grids [McAdams et al. 2011] or approximations with volume-like scaffolding [Zhou et al. 2005; Baran and Popović 2007; Zhang et al. 2010].

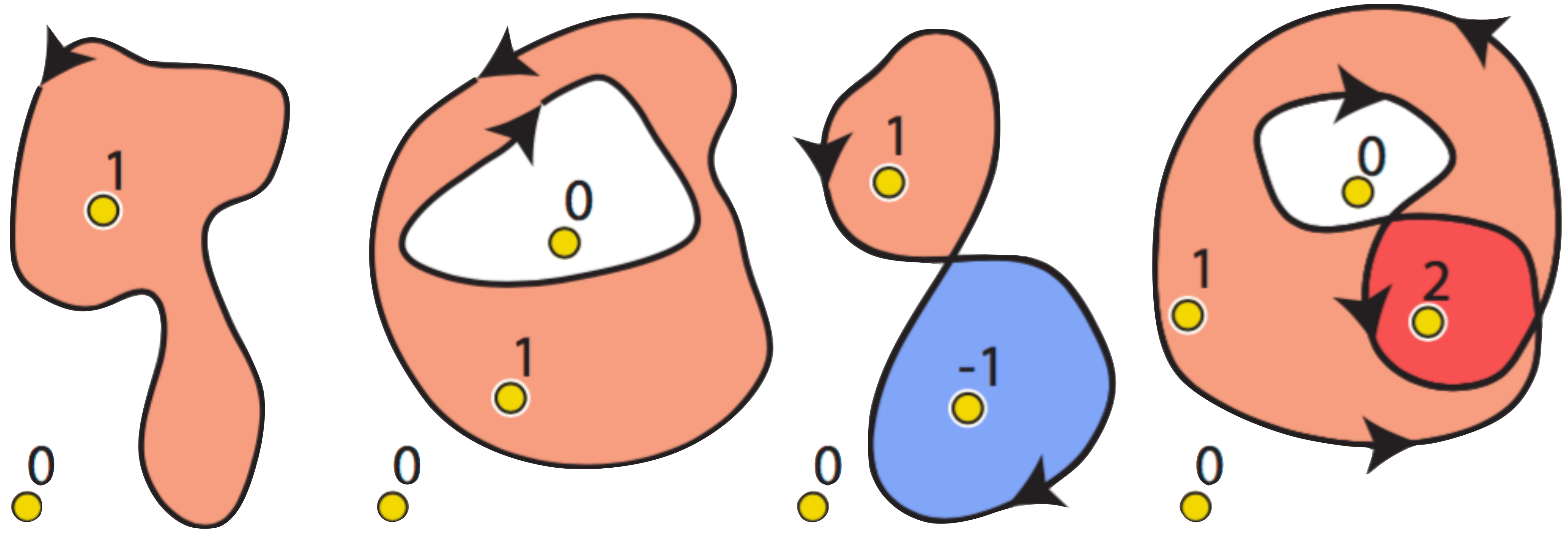
The aforementioned artifacts are common in man-made meshes, as these are the direct output of human creativity expressed through modeling tools, which very easily allow such artifacts to appear. Sometimes they are even purposefully introduced by the designer: for example, character meshes will typically contain many overlapping components representing clothing, accessories or small features, many of which have open boundaries (see Figure 2). Modelers

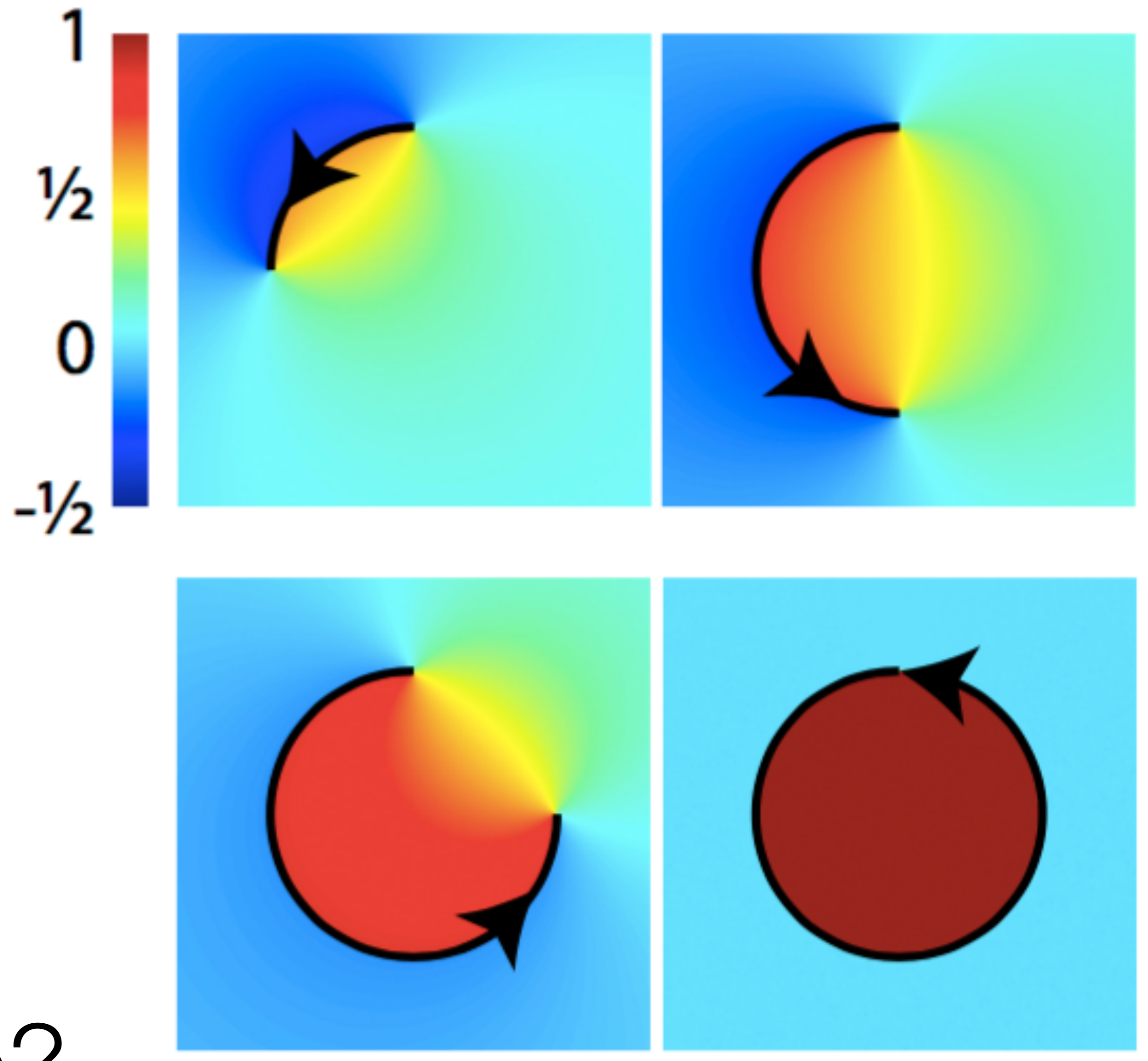
[1]Contrary to some authors' use of "conforming" to mean that every mesh edge is locally Delaunay, we use it simply to mean that the volume mesh interpolates to the boundary description.

Input mesh | Slice in output volume

Figure 2: *Each whisker, tooth and eye of the* Big SigCat *is a separate component that self-intersects the body.*

Generalized Winding Numbers

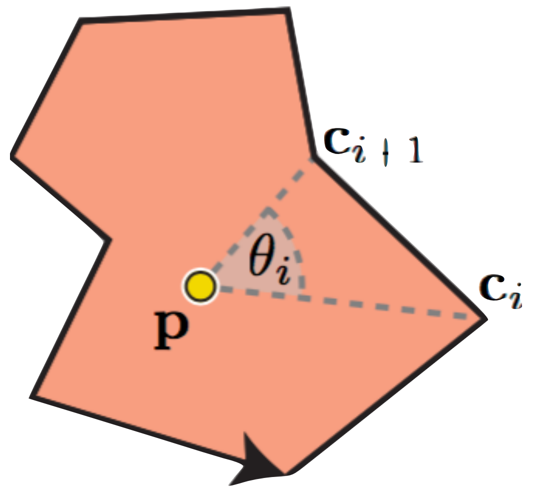




- Given arbitrary "line soup," how do we define inside outside? (Possibly with holes, self-intersections, non-manifold vertices, outliers…)

- Generalized winding number:

$$w(\mathbf{p}) = \frac{1}{2\pi}\sum_{i=1}^{n}\theta_i$$

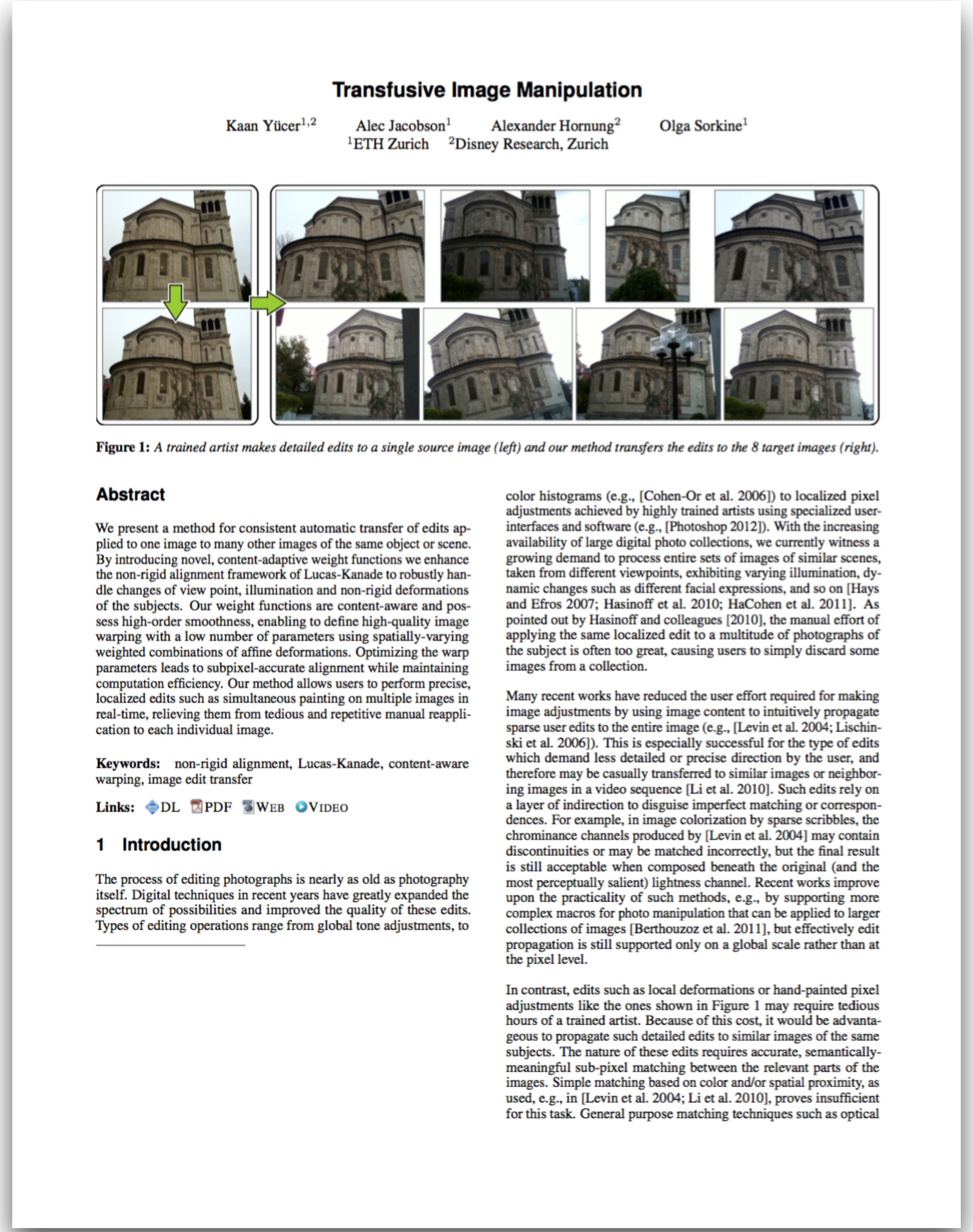


- Triangulate line soup, decide if each triangle inside or outside

- Solve graph cut problem for more robust segmentation.

NYU | COURANT

Transfusive Image Manipulation



Transfusive Image Manipulation

Kaan Yücer[1,2] Alec Jacobson[1] Alexander Hornung[2] Olga Sorkine[1]
[1]ETH Zurich [2]Disney Research, Zurich

Figure 1: *A trained artist makes detailed edits to a single source image (left) and our method transfers the edits to the 8 target images (right).*

Abstract

We present a method for consistent automatic transfer of edits applied to one image to many other images of the same object or scene. By introducing novel, content-adaptive weight functions we enhance the non-rigid alignment framework of Lucas-Kanade to robustly handle changes of view point, illumination and non-rigid deformations of the subjects. Our weight functions are content-aware and possess high-order smoothness, enabling to define high-quality image warping with a low number of parameters using spatially-varying weighted combinations of affine deformations. Optimizing the warp parameters leads to subpixel-accurate alignment while maintaining computation efficiency. Our method allows users to perform precise, localized edits such as simultaneous painting on multiple images in real-time, relieving them from tedious and repetitive manual reapplication to each individual image.

Keywords: non-rigid alignment, Lucas-Kanade, content-aware warping, image edit transfer

Links: DL PDF WEB VIDEO

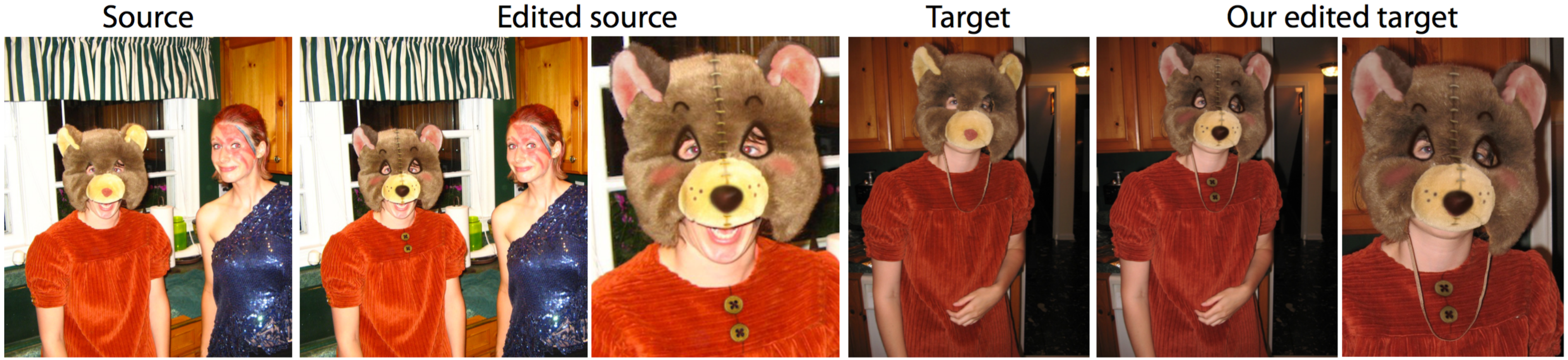
1 Introduction

The process of editing photographs is nearly as old as photography itself. Digital techniques in recent years have greatly expanded the spectrum of possibilities and improved the quality of these edits. Types of editing operations range from global tone adjustments, to color histograms (e.g., [Cohen-Or et al. 2006]) to localized pixel adjustments achieved by highly trained artists using specialized user-interfaces and software (e.g., [Photoshop 2012]). With the increasing availability of large digital photo collections, we currently witness a growing demand to process entire sets of images of similar scenes, taken from different viewpoints, exhibiting varying illumination, dynamic changes such as different facial expressions, and so on [Hays and Efros 2007; Hasinoff et al. 2010; HaCohen et al. 2011]. As pointed out by Hasinoff and colleagues [2010], the manual effort of applying the same localized edit to a multitude of photographs of the subject is often too great, causing users to simply discard some images from a collection.

Many recent works have reduced the user effort required for making image adjustments by using image content to intuitively propagate sparse user edits to the entire image (e.g., [Levin et al. 2004; Lischinski et al. 2006]). This is especially successful for the type of edits which demand less detailed or precise direction by the user, and therefore may be casually transferred to similar images or neighboring images in a video sequence [Li et al. 2010]. Such edits rely on a layer of indirection to disguise imperfect matching or correspondences. For example, in image colorization by sparse scribbles, the chrominance channels produced by [Levin et al. 2004] may contain discontinuities or may be matched incorrectly, but the final result is still acceptable when composed beneath the original (and the most perceptually salient) lightness channel. Recent works improve upon the practicality of such methods, e.g., by supporting more complex macros for photo manipulation that can be applied to larger collections of images [Berthouzoz et al. 2011], but effectively edit propagation is still supported only on a global scale rather than at the pixel level.

In contrast, edits such as local deformations or hand-painted pixel adjustments like the ones shown in Figure 1 may require tedious hours of a trained artist. Because of this cost, it would be advantageous to propagate such detailed edits to similar images of the same subjects. The nature of these edits requires accurate, semantically-meaningful sub-pixel matching between the relevant parts of the images. Simple matching based on color and/or spatial proximity, as used, e.g., in [Levin et al. 2004; Li et al. 2010], proves insufficient for this task. General purpose matching techniques such as optical

Transfusive Image Manipulation



- Determine transformation between regions of source and target image.
  - Minimize pixel difference over all transformations
  - Search subspace of “Linear Blend Skinning” maps
- Transfer edits using this transformation
- Warning: probably the most challenging project!

NYU COURANT